Amendments to the Specification:

Please replace paragraph 0001 with the following replacement paragraph:

[0001] The present invention relates generally to graphics processing and more specifically to the interleaving of [[with]]ALU operations with texture fetching operations.

Please replace paragraph 0003 with the following replacement paragraph:

[0003] FIG. 1 illustrates a prior art sequencing system 100. The system 100 includes a first arbiter 102, a second arbiter 102 and a third arbiter 103 and multiple buffers 104, 106, 108 and 110. In a typical embodiment, the buffers are first in and first out (FIFO) buffers. Each of the buffers 104-110 include multiple command threads, such as 112, 114, 116, 118 stored therein. Moreover, the system 100 is divided into resource divisions, such as an ALU resource division 120 and a texture fetch resource division 122. In the ALU resource division 120, the command thread 118 may be received from an input command 124 as selected by the arbiter 101. The command thread 118 may then be withdrawn from the reservation stations 104 and 108 for the purpose of being provided to an ALU (not shown) and the command threads within the texture fetch resource division 122 maybe withdrawn from the reservation stations 106 and 110 to be provided to a texture fetch processors (not shown).

Please replace paragraph 0004 with the following replacement paragraph:

[0004] In the prior art embodiments of FIG. 1, the first buffer 104 receives an input command 124 and outputs a completed command thread 126 to the second arbiter 102. In one embodiment, the command thread may include an indicator, such as a flag, indicating when the access to the ALU resources has been completed for the associated command. The arbiter

102 receives the input command 124 ~~126~~ and thereupon provides, in due course, the command thread to either an appropriate texture fetch buffer 110 or an ALU buffer 108. Thereupon, the steps are repeated where an output thread command 128 is provided to another ALU (not shown) or texture fetch processor (not shown) and returned to the buffer 108 or 110. The buffer 110 also produces the output 132 which is a command thread. The output 132 may be provided to another arbiter 103 to be provided further along the graphics processing pipeline.

Please replace paragraph 0019 with the following replacement paragraph:

[00019] In accordance with one embodiment to the present invention, FIG. 3 illustrates a pipeline vector machine 230 including a multiple ALU system 232, a buffer 234[[, a]] and sequencer logic 236, which may be an ALU resource. In one embodiment, the sequencer logic 236 receives a first thread 242, and a second thread 244 from the buffer 234, such that the logic 236 may perform simultaneous, interleaved execution of the command threads. Furthermore, the sequencer logic 236 is coupled to pipeline 240. In one embodiment, pipeline 240 may be an eight stage deep pipeline for providing vector analysis.

Please replace paragraph 0028 with the following replacement paragraph:

[0028] In this embodiment, the fields of the state bits, the control flow instruction pointer, the execution count marker, loop iteraters, call return pointers, predicate bits, are updated every time the thread is returned to the reservation station 302 or 304 based on how much progress has been made on the thread execution. It is also noted that in this embodiment[[s]], the GPR base pointer and context pointers are unchanged throughout the execution of the thread.

Please replace paragraph 0029 with the following replacement paragraph:

[0029] In one embodiment, the status bits include: a valid thread bit, a texture/ALU engine needed bit, a texture reads are outstanding bit and a waiting on texture read to complete bit. In this embodiment, all of the above status bit fields from the command threads go to the arbitration circuitry. Thereupon, the arbiter 306 selects the proper allocation of which command thread goes to the graphics processing engine 310 and which command thread goes to the ALU 308. In this embodiment, two sets of arbitration are performed: one for pixels, such as command thread 316 and one for vertices, such as command thread 322. ~~Although, t~~Texture arbitration requires no allocation or ordering as it is purely based on selecting the oldest thread that requires the graphics processing engine 310.